IN THE SPECIFICATION

Please amend the specification as follows:

At page 6, lines 12-13, please substitute the following paragraph:

a1 Figure 6a-b provides a high-level business representation of the configuration and operational flow of the architecture's reconciliation embodiment;

At page 7, lines 15-16, please substitute the following paragraph:

a2 Figures 20-22 is a detailed view of some of the database tables, indexes, and constraints of the reconciliation embodiment;

At page 16, line 18 – page 17, line 7, please substitute the following paragraph:

a3 The segment 120 of the diagram shows some of the automated data entry/integration facilities. These data integration facilities 120 are used by the architecture's applications to support the automated flow of business related data to and from the individual data sources for related clients. As can be seen by this representation, the structure generally contains some interface to client/source systems, which are external to the architecture and its related applications, but may be interfaced using a variety of facilities such as file or record handlers, direct message based connections into the architecture or, direct database level connections into the architecture. Using the connection facilities, data is pushed or pulled from the client systems and fed for processing through initial application decomposition and validation functions. These decomposition and validation processes utilize the reference library information in performing

A3 Cont

their individual functions and when complete, pass the related information or resulting business objects on for further processing within a given application of the architecture.

At page 17, lines 8-21, please substitute the following paragraph:

A4

The segment 130 of the diagram represents some of business-related processes or objects, which may be constructed utilizing the architecture to perform a given business function. The information shown in 130 relates to the reconciliation embodiment of the present invention and depicts a business process for matching the data arriving through the data integration facilities 120. The example also depicts the matching engines process for creating sets of related memory objects, data objects, display objects and report objects, based on the information it is receiving and the information contained in the related reference library objects. Data objects are also referred to as data storage objects in this application. This matching process 130 is representative of how the many processes of each of the set of processes required for any of the given application's of the architecture are constructed to utilize information flow, existing system objects, and reference library detail to achieve their individual business functions in a flexible manner. In matching engine 130, memory objects refer to the total set of in-memory objects in the architecture which includes both in-memory representations of the data objects, reference library objects, general configuration objects and/or the like.

At page 17, line 22 – page 18, line 8, please substitute the following paragraph:

A5

The segment 140 of the diagram represents some of the features which may be available through the user interface of any of the architecture's individual applications. These user interface features 140 are utilized to provide a dynamic interface for users to retrieve, display,

and update related business object and reference library information. The interface may also provide general features for exporting related business object or process data back to data sources or individuals in any number of different formats. In addition the interface may provide any range or reporting features for the related data. All of these related features are built on top of the individual application's reference library structure and utilize this object structure to maintain flexibility in how they display and manage the related classes of business information for each of the applications.

At page 19, lines 4-14, please substitute the following paragraph:

With the process of defining the processing requirements complete, the architecture's reference library functionality is examined for possible modification and/or enhancement in 202. The architecture's reference library may include a standard set of data abstraction, translation, and transformation services. The data definition object may include the ability to define the available data sources within an organization through user specified source system identifiers. The data definition object may also comprise the ability to specify the individual data elements available for each of the different source systems where the information may include the identifier, type, and format of the related data, the ability to load the related information in an automated or semi-automated fashion with the assistance of a data dictionary within the organization, the ability to specify for each data source a related database to use during the automated extraction of information from that system, and the ability to specify for each user-defined field identifier a related database field identifier as shown in box 202.

At page 24, line 20 – page 25, line 9, please substitute the following paragraph:

Figure 5 describes the process of a user interacting with the applications facilities to review the business data, perform additional processing on the related data, adjust or modify the data, and report or communicate any required status information regarding the data. These facilities are provided to users through the applications web-based interface, in 501. The system may provide a screen or sets of screens for retrieving the related business data grouped, organized, and/or filtered by the status of one of the primary data storage objects. This embodiment is demonstrated in detail within the context of a data reconciliation system where data records are displayed in relation to their related data group object 1661. The system provides facilities for filtering the related information on fields such as business data, system date, and data group state in 502. From the applications' data review screens, the user can trigger a number of system related processes on the data being viewed. For example, in the case of the reconciliation embodiment, a user may select individual data groups and perform tasks such as manually closing the group or re-reconciling the group.

At page 28, line 13 – page 29, line 6, please substitute the following paragraph:

Figure 8 details the reconciliation manager's operational steps, supporting the ongoing receipt and/or extraction of data and its submission for initial processing, and parallels the steps initially discussed with regards to figure 4. This process begins with data receipt and extraction in 801. In this embodiment, the schedule of the information flow may be driven by the characteristics of the individual reconciliation. As before, any of the architectures integration facilities can be utilized. After the data receipt, the decomposition process begins with the validation of header information, such as client identifier, reconciliation identifier, and system identifier of the individual record. Once the validation of header information is complete, the

match key string is created based on key field objects defined for the underlying reconciliation system. Next, the data comparison structure is interpreted and the individual data comparison values are computed and stored in related data storage objects in step 802. The next step in decomposition is the processing of the information field objects against the data provided, and creating the related information data storage objects in step 802. Next, this new item is passed on for matching, the process, which groups the item with existing data in the system. Subsequently, if the resulting data group is completely matched and the reconciliation is structured to reconcile data real time, the data group is passed to the reconciliation process in step 803.

At page 42, lines 3-16, please substitute the following paragraph:

Figure 14 contains additional objects which are part of the set of objects supporting Reconciliation Manager's reference library functionality. The data compare attribute (data compare attrib) objects 1401 is a set of objects that control the individual processing characteristics of each of the different data comparisons for an individual reconciliation. According to one embodiment, a given reconciliation can have any number of comparisons with each comparison utilizing exactly one data compare attribute object 1401. These data compare attribute objects 1401 are core to providing the intelligence/processing characteristics of a reconciliation's individual data comparisons and, control precisely how the application determines which data elements, from matched data items of each of the different systems of a reconciliation, can be considered equal. The attribute objects 1401 are uniquely identified in the system using a client identifier 1402 that is inherited from the parent object, reconciliation identifier 1403 that is inherited from the parent object, compare identifier (compare identifier)

1404 which is a system generated field that assigns a numeric identifier to the individual comparison as it is created by the user.

At page 43, line 16 – page 44, line 13, please substitute the following paragraph:

The set of comparison type objects (comparison type) 1421 contains the different comparison types supported by the system. This object/field controls the values which can be entered into the comparison field (cmpcmptype) 1409. The set of available tolerance processing types objects (tolerance type) 1431 provides the available tolerance processing options a user can select within the system. These values are stored in field (cmptolprctype) 1432. The reconciliation system data compare (reconciliation system data compare) 1441 objects define additional characteristics of the data comparison process for each system and data comparison of a reconciliation. The reconciliation system data compare objects are uniquely identified within the system using client identifier 1442, inherited from parent object reconciliation system 1371, reconciliation identifier 1443, inherited from parent object reconciliation system 1371, system identifier 1444, inherited from parent object reconciliation system 1371, compare identifier 1445 that is inherited from the related data compare attribute object 1401, ignore space 1446 that is inherited from the related data compare attribute object 1401, compare data type 1447 that is inherited from the related data compare attribute object 1401. In addition, according to one embodiment, a user selected field that determines how the system will combine multiple values from the given system and comparison into an ultimate value for reconciliation (cmpoprt) 1448 may be part of the reconciliation system data compare objects. The values available for cmpoprt 1448 are the full set of cmpoprt options stored in the set of reconciliation comparison operator objects. This cmpoprt value 1448 in conjunction with data type 1447 will determine how the

a10 cont

ultimate comparison value is derived. For example, a numeric data type could use the average setting to compute an average value for all the fields which are part of the particular comparison from the given system not clear return.

At page 52, line 3 – page 53, line 2, please substitute the following paragraph:

a11

The item information element also contains a field which holds the data for its associated field identifier (flddatchar) 1734. This data is extracted from the related message and placed in flddatchar 1734 as part of the message decomposition process. Item compare element objects (item compare element) 1741 are used by Reconciliation Manager to store and manage the derived comparison values for the related reconciliation item parent object 1701. These item compare elements are uniquely identified in the system using item identifier 1742 that is inherited from the parent object, and a compare identifier 1743 that is inherited from its related reconciliation system data compare object 1441. The object 1741 also contains a system derived value which indicates if a comparison value is expected for the related reconciliation system data compare objects 1441 (cmpactive) 1744, the comparison's data type (cmpdattype) 1745 that is inherited from the related reconciliation system data compare objects 1441, a [list of] list of field identifiers obtained from the related set of reconciliation data field objects 1471 used in computing the value (cmpfldids) 1746, a field which stores a user generated correction value or note for the individual comparison element value (cmprefval) 1747, a system generated string based display version of individual compare value (cmpdispval) 1748, the system generated derived character value for the comparison element (cmpvalchar) 1749 which is used only if cmpdattype is "string", the system generated derived numeric value for the comparison element (cmpvalnumb) 1750 that is used only if cmpdattype is "number", the system generated derived

a11 cont

date time value for the comparison element (cmpvaldatetime) 1751 that is used only if cmpdattype is "date", a system generated string containing the data values which went into computing the related derived value for the element (cmpfldvals) 1752, and a system managed field containing a numeric status indicator for the individual status element (cmpstat) 1753.

At page 53, line 10 – page 54, line 18, please substitute the following paragraph:

a12

The archive data object also contains a client identifier 1803, reconciliation identifier 1804, key identifier 1805 each inherited from the related data group object 1661, original group identifier (origgrpid) 1806 that is inherited from group identifier 1665, last business date update (lstbusdatupd) 1807, lasts system date update (lstsysdatupd) 1808, number of system's unmatched (noofsysunmched) 1809, group status (grpstat) 1810, wherein each of these fields is inherited from the related data group object 1661. The archive data object also contains original absolute data group identifier (origabsgrpid) 1811 that is inherited from absolute data group identifier (absdatgrpid) 1670, original data group notes (origgrpnotes) 1812 that is inherited from notes 1671, original data group has error (origgrphaserror) 1813 that is inherited from has error field (haserror) 1672, original data group error message (origgrperrmessage) 1814 that is inherited from error message 1673, data group match queue data (grpmchquedata) 1815 which is a system generated value containing a condensed version of all of the data group objects related group match queue objects 1641, data group comparison data (datgrpcmpdata) 1816 which is a system generated value containing a condensed version of all of the data group objects related data group compare objects 1691, reconciliation minimum business date (recitemminbusdate) 1817 which is a system generated value containing the minimum business date of all related reconciliation items 1701, reconciliation item maximum business date (recitemmaxbusdate) 1818

which is a system generated value containing the maximum business date of all related reconciliation items 1701, reconciliation item minimum system date (recitemminsysdate) 1819 which is a system generated value containing the minimum system date of all related reconciliation items 1701, reconciliation item maximum system date (recitemmaxsysdate) 1820 which is a system generated value containing the maximum system date of all related reconciliation items 1701, reconciliation item original text (recitemorigtext) 1821 which is a system generated value containing a condensed version of all of the data group objects related reconciliation item objects 1701 related item fields 1703, reconciliation item data (recitemdata) 1822 which is a system generated value containing a condensed version of all of the data group objects related reconciliation item objects 1701, reconciliation item information element data (reciteminflmntdata) 1823 which is a system generated value containing a condensed version of all of the data group objects related reconciliation item objects 1701 related item information element objects 1731, reconciliation item compare element data (recitemcmplmntdata) 1824 which is a system generated value containing a condensed version of all of the data group objects related reconciliation item objects 1701 related item compare element objects 1741.

At page 54, line 19 – page 55, line 9, please substitute the following paragraph:

Figure 19 shows a file reader object (file reader) 1901 that is used by the application to manage the reader processes for a given client. These reader processes are used to retrieve data record files for the client from the server and submit the reconciliation text messages contained in these files to Reconciliation Manager for processing. The file reader objects are uniquely identified in the system using client identifier 1902 that is inherited from the parent object 1211, and the file reader identifier (frdrid) 1903 which is a system generated sequence number for the

individual file reader object. The object also contains a user enter value specifying the directory location where a related reader process will look for files on the application server (finputdir) 1904, a user enter value specifying the directory location where a related reader process will generate output files on the application server (foutputdir) 1905, a system managed field indicating if the associated reader process is currently active for the file reader object (active) 1906, a system managed field indicating if the associated reader process is currently in the process of shutting down for the file reader object (sdinprog) 1907, and a system managed field indicating any error messages generated by an associated reader process (errmsg) 1908.

At page 57, lines 5-12, please substitute the following paragraph:

Other points of interest in relation to the above mentioned figures includes the use of relational table constraints and the method of persisting data between the objects and the tables. Table constraints are used between the tables to ensure the continuous integrity of the tables and related objects data. For example there exists a constraint between client table (rhcl) 2101, and the file reader table (rhfilereader) 2103. This constraint ensures that each record in the file reader table 2103 contains a clid values which exists in the client table 2101. In terms of persisting data between objects and tables this is achieved using industry standard techniques supported by the EJB standard and the web server's infrastructure.

At page 60, lines 11-15, please substitute the following paragraph:

During the creation process, all leading and trailing spaces are removed from the system identifier and field identifier in 2409. Once the object creation process is complete, the system creates a record for the object in table rhsysfld 2008 and returns the object to the calling adapter

program rhsysfldform 1120 in 2410. The adapter program then refreshes the user's screen and display the information entered in 2411.

At page 60, line 20-page 61, line 23, please substitute the following paragraph:

The create reconciliation definition process in figure 25 begins with a user selecting the "Reference Library" menu option then selecting the "Reconciliations" menu option presented by the main window interface 1103. Making these selections will call adapter program rhclrecform 1122 and presents the user with an additional "Add Rec" menu option. On making this selection, rhclrecform 1122 calls rhrecaddform 1125 and the add reconciliation process begins. Adapter program rhrecaddform 1125 presents the user with a screen for entering a reconciliation identifier and a reconciliation description. After this information is entered by the user and submitted we begin our processing with step 2501. Adapter program rhrecaddform 1125 checks the existence and length of the reconciliation identifier and reconciliation description fields in 2502. If the information provided is valid, in 2503, adapter program rhrecaddform 1125 calls rhrecform 1124, which retrieves the client object 1211 from the existing user session in 2505. If the information entered is not complete or correct, the caller is alerted and asked to fix the data provided in 2504. With client object the adapter program then calls the client object's 1211 addrec method passing as parameters the reconciliation identifier and the reconciliation description. Using the client identifier 1212 from client object 1211 and the information provided, the addrec method attempts to create a new reconciliation definition object 1341, in 2506. If the provided reconciliation identifier is unique within the given client objects' existing set of reconciliation objects 1341, then the reconciliation definition object is created in 2507. However, if the reconciliation identifier is not unique or some other unforeseen error occurs, the

addition process is abandoned and the caller is notified in 2508. If the object creation is successful, then client identifier 1342 will be set to client identifier 1212, reconciliation identifier 1343 will be set to the reconciliation identifier provided, reconciliation description 1344 will be set to the description provided. All other variables 1345 through 1356 will be set to either "0", "null", or "false", depending on the data type of the individual fields. The setting provided on object creation for variables 1345 through 1356 are only temporary as the related values will be set/modify during later system processes. During this creation process, all leading and trailing spaces are removed from both the reconciliation identifier and reconciliation description in 2509.

At page 62, lines 11-15, please substitute the following paragraph:

The process begins with a user selecting the "Reference Library" menu option, then selecting the "Reconciliations" menu option, presented by the main window interface 1103. Making this selection calls adapter program rhclrecform 1122 which present the user with a second screen. On this second screen the user can highlight a reconciliation and select the "Add System" option, which calls rhrecsysaddform 1127 and begins the process.

At page 64, lines 1-8, please substitute the following paragraph:

If the object creation is successful then client identifier 1372 will be set to client identifier 1342, reconciliation identifier 1373 will be set to reconciliation identifier 1343, system identifier 1374 will be set to the system identifier selected originating from the system definition object's system identifier field 1303, ignore character space (ignspace) 1375 will be set to the related derived value, and ignore character case (igncase) 1376 will be set the related derived value derived from user entered setting for the process. During this creation process, all leading and

A-18 cont

trailing spaces are removed from the client identifier, reconciliation identifier, system identifier and the data retrieval and update information will be set as indicated by the user in 2609.

At page 65, line 18 – page 66, line 2, please substitute the following paragraph:

A-19

If the system field identifier is selected and the other information of the form is set properly adapter program rhreckeyaddform 1129 calls rhreckeyform 1128 which retrieves the client object 1211 from the existing user session, and using the client object's get reconciliation (getrec) method retrieves the reconciliation object 1341 for the selected reconciliation 2705. Then using the reconciliation object's get reconciliation system (getrecsys) method, the routine retrieves the reconciliation system object 1371 for the selected system 2705. If a system field identifier selection is not made, the caller is alerted and asked to correct the problem in 2704.

At page 70, lines 1-14, please substitute the following paragraph:

A20

Adapter program rhrecdatfldaddform 1132 presents the user with a screen for selecting a field identifier from the set of system field objects 1311 belonging to the selected system definition 1301. This list of fields is obtained by retrieving client object 1211 from the existing user session and using the client object's get system (getsys) method to return the system definition object 1301 for the selected system identifier. Then, using the client object's get reconconciliation (getrec) method the reconciliation object 1341 is obtained and, using the reconciliation object's get reconciliation comparison attribute (getreccmpatrib) method the data compare attribute object 1401 is retrieved. Then, using the selected system definition object's 1301 list system fields by field type (listsysfldsbyfldtype) method and the data compare attribute object's comparison data type (cmpdattype) field 1406, as a parameter, a list of system field

objects 1311 of the appropriate data type is retrieved. Other options included on this screen include the ability to specify how fields from the same system of a comparison should be combined. By selecting a field for addition and submitting the information, the user begins the process in 2901. Adapter program rhrecdatfldaddform 1132 then checks the selection, 2902.

At page 77, lines 1-9, please substitute the following paragraph:

The single reconciliation message process of figure 32 begins with one of the calling methods initiating this process with a reconciliation message as a parameter in the form of a string in 3201. On receiving this call, the process checks that the message string is not empty in 3202. If the reconciliation message is empty, an alert message is printed on the server console in 3203 and the process ends in 3204, otherwise the processing continues with a call to create a new reconciliation item object in 3205 described in figure 33. The create reconciliation item process either returns a valid reconciliation item object 1701 or throws an exception. In the case of an exception being generated, this is handled in step 3210 which is described later and is used to manage all exceptions received by the single reconciliation message process.

At page 77, line 16 – page 78, line 5, please substitute the following paragraph:

After receiving a decomposed reconciliation item 1701 back from process of figure 34, the program calls the match process with the reconciliation item 3207. The match process will place the reconciliation item in the appropriate data group 1661 based on the combination of match key value 1710, system identifier 1706, system identifier 1707, and reconciliation identifier 1708. This process is detailed and described beginning in figure 40. If this match process 3207 completes and returns a data group object 1661 then two events have occured.

First, the reconciliation item must be part of a data group that is completely matched (i.e., contains at least one reconciliation item 1701 from each of the systems contained in the reconciliation), and is ready to have its data elements reconciled. Second, the particular reconciliation object 1341 for this reconciliation item 1701 must have its reconcile data group real time flag (recrealtime) 1350 set to true. This option for supporting the real-time reconciliation of data groups is a critical feature and allows Reconciliation Manager to generate real-time notification messages back to individual users or systems as data breaks are found between the data records submitted for processing.

Q22 cont

At page 78, lines 6-12, please substitute the following paragraph:

Q23

On receiving a data group 1661 back from the match process in 3208, the application will make a call to the reconciliation process described in figure 45, in step 3209. This reconciliation process of figure 45 goes through each of individual comparisons constructed for the reconciliation and compares the data elements from each of the individual system. This process of figure 45 results in a detailed analysis and understanding of which of the data elements provided from each system in the particular data group is potentially in error. If no data group 1661 is returned and no exceptions have occurred, in 3210, then the process ends in 3211.

At page 80, lines 1-6, please substitute the following paragraph:

Q24

Then the header validation process is called in step 3403. This header validation process is detailed in figure 36. The processes primary task is to extract and check the primary details of user identifier, system identifier, and reconciliation identifier from the message string provided. These details are then used to map the reconciliation item 1701 back to the individual

Q24 Cont components of the applications reference library, which guide the remaining decomposition processes.

At page 80, lines 7-10, please substitute the following paragraph:

Q25 Then, the build key process is used to construct a match key value 1710 for the reconciliation item object 1701 in 3404. The build key process is described in figure 37. The resulting match key value is used in later processes to determine which data group object 1661 this item belongs to.

At page 80, line 20- page 81, line 3, please substitute the following paragraph:

Q26 Once the set of processes is complete, the reconciliation item 1701 is returned to the calling process in 3407 and the decomposition process ends in 3410. If, however, any processing error is generated during this decomposition process the routine jumps to step 3408 and then 3409. In this error case, any subsequent process after the process that generated the error is not called, and a reconciliation item object is not returned to the caller. Regardless of the error type, the application will return a detailed decomposition exception back to the calling process in 3409 and end the process in 3411.

At page 81, lines 8-16, please substitute the following paragraph:

Q27 The process begins on receipt of the call in 3501 and continues to iterate through steps 3503 through 3505 until the entire message has been processed in 3502. The process works by searching the given string sequentially for the start of a field token i.e. "<" followed by ">", in 3503, in this process any text between these symbols is taken as the token identifier such that a

a27 cont

sample of a complete starting token could look like "<System identifier>". Then, the process continues searching sequentially for the corresponding end token, such as "</System identifier>" in 3504. The process then extracts the portion of the string stored between the start and end tokens and places this in the hash table with a key value equal to the token identifier-field identifier such as System identifier in 3505.

At page 81, lines 17-22, please substitute the following paragraph:

a28

The process checks for several types of processing errors as represented in 3506. For each error that occurs, a detailed parse XML exception will be sent back to the calling routine in 3508 and the process will be terminated in 3509. Duplicate token identifier are considered an error as well as any missing end of token identifiers i.e. ">"[, and end token identifiers]. Once complete, if no errors have occurred, the process ends normally in 3507 and the hash table is available for use.

At page 83, lines 14-16, please substitute the following paragraph:

a29

Figure 37 details the build key process, which is used too create a match key string for the reconciliation item 1701 from the data provided and the reference library information for the particular reconciliation identifier 1708 , system identifier 1707, and client identifier 1706 combination.

At page 83, lines 17-22, please substitute the following paragraph:

a30

The build key process begins with receipt of the call containing an item object 1701 as the parameter in 3701. The build key process first initializes several local key value variables to

represent a new and empty match key strings in 3702. Then using the reconciliation system object 1371, as set in the header extraction process of figure 36, the application calls the reconciliation systems (listrecsyskeyflds) method, retrieving a set of reconciliation key field objects 1381 for the reconciliation system 1371 in 3703.

At page 84, lines 1-7, please substitute the following paragraph:

For each key field object 1381 in the set, the application performs the following steps 3704. First, the field identifier 1385 is obtained from the reconciliation key field object 3705. Then the hash table is checked to ensure it has a value for this field identifier; if no value exists in 3706, the process jumps to step 3717, generates a build key exception and terminates at step 3718. If the value exists, the value's expected type is checked in 3707 using the field type value from (fldtype) 1387. If the type is not a date, the data value from the hash table is appended to the existing local key value variable 3709 and the process returns to step 3704.

At page 85, lines 5-9, please substitute the following paragraph:

The build key process contains general exception management as represented in step 3715. This allows the process to respond to any unexpected errors such as date conversion problems. If any error occurs during the build key process, the application jumps to step 3715, throws a build key exception 3717, and terminates with step 3718. If step 3714 is completed successfully, the flow move through step 3715 and end the process normally at step 3716.

At page 85, lines 16-22, please substitute the following paragraph:

The process begins with the receipt of a method call containing the reconciliation item object 1701 as a parameter in 3801. The next step in the process is to initialize primary variables to hold the comparison type information and a list of field identifiers processed in 3802. Then using the reconciliation system object's 1371 list reconciliation system data compare objects (listrecsysdatcomps) method, a list of the reconciliation system data compare objects 1441 for the given reconciliation system as set in the header extraction process of figure 36, is obtained in 3803.

At page 86, lines 1-16, please substitute the following paragraph:

Then starting with the first data compare object 1441, representing an individual data comparison point, and processing all compare objects 1441 in the set, the application performs each of the following step in 3804. The application resets the local variables which will be used to hold, the computed value for the current comparison, the field identifiers used in creating the value, the type of the value, and the format of the value 3805. Once these variables are reset, the application retrieve the set of reconciliation data field objects 1471 using the data compare object's 1441 list reconciliation system data fields (listrecsysdatflds) method 3806. The application then performs the following set of processes for each of the data fields objects 1471 in the set in order to derive a single comparison value for the reconciliation comparison and the system. Next, the application retrieves the field identifier using field 1476 of the current reconciliation data field object 1471. Then the application checks that the shared decomposition hash table contains a value for the field identifier in 3809. If no value exists, the process jumps to step 3819, generates a build data exception back to the caller and terminates with step 3820. If the value exists the application retrieves the value from the hash table for processing in 3810.

Q34 CONT

The application then appends the field identifier value to the local variable containing list of field identifier used to compute our final value 3811.

At page 87, line 16 – page 88, line 4, please substitute the following paragraph:

Q35

If this process is successful the item compare element 1741 is instantiated with the following details. Item identifier 1742 is set to item identifier 1702, compare identifier 1743 is set to compare identifier 1445, compare active 1744 is set to the true or false indicator provided, compare data type 1745 is set to comparison data type 1447, compare field ids 1746 is set to the derive list provided, compare reference value 1747 is set to null, compare display value 1748 is set to the derived display value, one of compare value char 1749, number 1750, date 1751 will be set to their related value provided. Note, only one of these fields will actually be used and this will depend on the value of the data type information. Compare field values 1752 will be set to the derived vale provided, and the compare element status field 1753 is set to zero representing a new element. On successful completion of the instantiation process a record for the object is created in table rhrecitemcmplmnt 2211 and the process moves back to step 3804 to derive data for the next ~~comparison~~.

At page 88, line 14 – page 89, line 7, please substitute the following paragraph:

Q36

The process begins with the receipt of a method call containing the reconciliation item 1701 as a parameter in 3901. The process then retrieves the set of reconciliation information field objects 1501 for the given reconciliation system object 1371 as set in the header extraction process of figure 36. This is achieved using the related reconciliation system object's 1371 list reconciliation system information fields (listrecsysinffIds) method in 3902. Then the application

goes through this set of information field objects 1501, and as long as there are more objects 1501 to look at, the application performs the following set of actions in 3903. The process gets the information field objects field identifier 1505, in 3904. Then, the build information process checks the field identifier has a corresponding value in the in memory hash table in 3905. If no value exists, a build information exception is returned to the caller and the process terminates in steps 3905, 3910, 3911; otherwise, the process attempts to create a new item information element object 1731 using the reconciliation item's 1701 add reconciliation item information element (reciteminflmnt) method passing as parameters the field identifier 1505, and the value retrieved from the hash table in 3906. On creation, the item information element is instantiated and item identifier 1732 is set to item identifier 1702, field identifier 1733 is set to field identifier 1505, and the field data string 1734 is set to the value from the hash table in 3907. Once complete, a record is created for the new object in table rhreciteminflmnt 2210.

A36 Cont

At page 94, lines 4 - 8, please substitute the following paragraph:

A37

If a system match queue object is not found in 4203, the process jumps to step 4204 that calls the add data group process and returns the data group object 1661, which was returned from the add process, then proceeding to step 4218 for process completion. If a system match queue object 1631 is found, the process then uses the system match queue's get group match queue (getgrpmchque) method to retrieve the group match queue object 1641, in 4205.

At page 95, lines 3 – 8, please substitute the following paragraph:

A38

The system then checks if the data group is completely matched in 4213 by looking at the value of the number of systems un-matched 1668 stored on the group object. If this value is not

zero then the process moves to step 4217 returning null to the caller. If the value is zero then the process will set the reconciliation data objects data group counters 1611 through 1612 to reflect that there is one less data group unmatched and one additional data group pending reconciliation. Then the status of the data group held in field 1669 is set to "1" indicating the group is fully matched pending reconciliation in 4214.

a38 cont

At page 95, lines 9 – 15, please substitute the following paragraph:

a39

After completing this step, the system checks the reconciliation's real-time setting 1350, and if this is true in 4215 then the data group object is returned to the calling process which will pass this object on for reconciliation in 4216. Otherwise the process returns null to the caller and leaves the user to complete the data group's reconciliation process in 4217. If any errors occur during this process, they are trapped at step 4218 which will proceed to step 4219, generating a group new exception back to the caller and terminating the process in 4220. If no errors have occurred, the process will simply terminate normally at step 4220.

At page 97, lines 4-17, please substitute the following paragraph:

a40

Using values from the parent object and the business date value provided the new data group object 1661 is created in 4404. After creation the object instantiation process begins by retrieving the related sequence generation object 1951 for the key value of "absdatgrpid". Then calling the object's get sequence number (getseqnumber) routine which increments the sequence number in the object by one and then returns the starting sequence number. This value is then used to set the data group's absolute data group identifier 1670. In this process, client identifier 1662 is set to client identifier 1602, reconciliation identifier 1663 is set to reconciliation

identifier 1603, key identifier 1664 is set to key identifier 1604, group identifier 1665 is set to last group identifier 1608, last business date 1666 is set to the business date provided, last system date 1667 is set to last system date 1606, the number of systems unmatched 1668 is set to number of systems 1607, group status 1669 is set to zero, the notes field 1671 is set to null, the has error indicator 1672 is set to false, and the error message field 1673 is set to null. Once instantiation is complete a record is created for the object in table rhdatgrp 2109 and the record is returned ready for use in the initial process in 4405.

A46 cont

At page 98, lines 10-19, please substitute the following paragraph:

A41

Step 4413 uses the add system match queue (addsysmchque) method of the reconciliation data object 1601 to return a system match queue object 1631 for the system identifier 1707 of the current reconciliation system object. This process will simply return the object 1631 if it exists or will create the object, instantiate it using information from its parent object and the system identifier provided. This process creates a record for the object 1631 in table 2108, and then returns the object. Using the system match queue object's 1631 add group match queue (addgrpmchque) method a group match queue object 1641 is created using details of the parent object and the group identifier 1665, creating a record for the object 1641 in table 2110, in 4414. This process results in a set of group match queue object 1641 being created to instruct the application as to which source system have not yet provided an individual reconciliation item 1701 for the new data group 1661. On completing this process the application proceeds to step 4419 returning null to the caller.

At page 98, lines 20-23, please substitute the following paragraph:

If, however, the data group is completely matched in step 4409, the application proceeds to step 4415, setting the reconciliation data objects data group counters 1611 and 1612 to reflect that there is one less data group unmatched and one additional data group pending reconciliation. Then the status of the data group held in field 1669 is set to "1" indicating the group is fully matched pending reconciliation 4416.

At page 99, lines 1-7, please substitute the following paragraph:

After completing this step, the system checks the reconciliation's real-time setting 1350, and if this is true in 4417 the data group object 1661 is returned to the calling process which will pass this object on for reconciliation in 4418. Otherwise the process returns null to the caller and leaves the user to complete the data group's reconciliation process in 4419. If any errors occur during this process they are trapped at step 4420 which will proceed to step 4421 generating a add data group exception back to the caller and terminating the process in 4422. If no errors occur, the process terminates at step 4422.

At page 102, lines 9-20, please substitute the following paragraph:

Fourth, in 4517, the process iterates through the reconciliation items 1701performing the following actions for each item which requires comparison: (1) select the individual reconciliation item 1701, in 4518; (2) retrieve the related item compare element 1741 using the reconciliation item's 1701 get reconciliation item compare elements (getrecitemcmplmnt) method and the compare identifier 1404, in 4519; (3) set the element's comparison status (cmpstat) 1753 to indicate reconciled in 4520; (4) retrieve the value from the appropriate compare element field 1749 through 1751, based on the comparison type; (5) combine the value

with the existing comparison value for the key group (it should be noted that this can be done through a variety of options including addition, averaging, and concatenation in 4521); (6) if the application is not in key group mode or, this is the last reconciliation item in the set or, the next reconciliation item belongs to a different key group in 4522 then proceed to step 4523 otherwise the application goes back to step 4517.

At page 103, line 15 – page 104, line 2, please substitute the following paragraph:

Once this process has been performed for each of the related data compare attribute objects 1401, the system moves from step 4511 to step 4529. At this point, the process updates the reconciliation data objects 1601 data group status counters reducing the number of matched pending reconciliation data groups (noofmchpndrecdatgrps) 1612 by one, and incrementing either number of reconciled with data breaks (noofrcldwthbrkdatgrps) 1613 or number of reconciled with no data breaks (noofrcldnobrkdatgrps) 1614 counters by one based on the derived reconciliation status of the group, in 4529. The process then set the status as required on the data group 1661 in field group status (grpstat) 1669. Then, the process will go through each of the related reconciliation item objects 1701 and set their individual item status (itemstat) fields 1712, and match status (matchstat) fields 1713 to indicate the status of the group either reconciled or data breaks in 4530.

At page 108, line 5– page 109, line 21, please substitute the following paragraph:

Figure 49 describes the range of features which exist in the application for users to retrieve, review, and update the data submitted and processed for their related reconciliations. This functionality is accessible to a user of the application via the user interface and the

processed data and reconciliation results menu options in 4901. Selecting these options will utilize the interface adapter programs 1139-1145 to provide a number of functions. On initial selection, the user interface is displayed for the first reconciliation in the list of user reconciliation and the default selection criteria of the related adapter program in 4902. At any point, the user can reset the selection option and after making the necessary changes, the screen will be refreshed. Some of the selection options provided include the ability to select a reconciliation and for the reconciliation set which data group types are desired, unmatched, matched pending reconciliation, reconciled with data breaks, reconciled with no data breaks, manually closed, and manually ungrouped. Also included is the ability to filter data groups on date range either by system processing date or by business date. A further option allows a user to specify which data comparisons will be looked at for any of the reconciled data groups in 4903. On completing a selection, data group and related object information is retrieved and presented to the user in 4904. The system then gives a range of options to the user. For example, the user can initiate the reconciliation process for the selected reconciliation object 1341 which will retrieve all data group objects which are pending reconciliation and will go through the set of object, submitting each object for reconciliation using the reconcile process described in figure 45, in 4905. The user can select a given data group in 4906 by highlighting the group on the screen at which point the system provides a number of options, such as the ability to double click on a group open a new window displaying information regarding the data group 1661, the data groups related queue information 1641, the group's reconciliation items 1701 and related item compare and information elements 1731, 1741. In addition, the ability to manually close a data group and prevent it from being used in further matching or reconciliation is provided, which is achieved by pre-determined key stroke(s). The process removes all of the

group's related queue objects 1641 and data group compare objects 1691, then the appropriate status counter for the parent reconciliation data object is updated 1601, 1610 - 1616 and the status of the group status (grpstat) 1669 is changed. Another option relates to the ability to reset a data group which will bring it back to either the unmatched state or the matched pending reconciliation state, which is achieved by pre-determined key stroke(s). The process runs the close group process on the data group, resets the business and system dates 1666, 1667, creates any required system match queue and group match queue objects for the related list of reconciliation system object 1371 not represent by reconciliation items 1701 currently allocated to the data group, reset status indicators of the reconciliation data object 1601 and the sets the data group's group status (grpstat) 1669. Furthermore, the ability to submit an individual data group for reconciliation by pre-determined key stroke(s) is provided. This process closes the data group, resets the data group, and if the status of the group is then matched pending reconciliation the group is submitted for reconciliation using the reconcile process of figure 45.

At page 109, line 22– page 110, line 19, please substitute the following paragraph:

As another feature, the ability to move a data group into the online archive is provided, which is achieved by a pre-determined key stroke(s). This process uses the move data group to archive process of figure 53 for transfer the selected data group 1661 to the archive. If the user has selected a combination data group in 4906 and a reconciliation item in 4907, then the application provides the ability to remove the selected item from its current data group and place it in the selected data group, which is achieved by pre-determined key stroke(s). This move reconciliation item process will close each of the data groups, set the reconciliation items related group identifiers 1711, 1717, to the new data group identifiers 1665, 1670, reset each of the data

groups, and delete the data group from which the item was removed if that group contains no other reconciliation items 1701 in 4911. On selecting only a reconciliation item the system provides the ability to move the selected item to a system managed data group called the ungroup data group. This functionality is used to pull individual items out of the application's processing structure, which is achieved by pre-determined key stroke(s). The un-group reconciliation item process closes the data group, gets or creates the reconciliation data object 1601 for the reconciliation item's match key value 1710, gets or creates the special un-group data group for the reconciliation data object, set reconciliation items related group identifiers 1711, 1717, to the new data group identifiers 1665, 1670, reset each the original data group, and delete the data group which the item was removed from if that group contains no other reconciliation items 1701 in 4910. Other core functionality provided by this interface is the ability to set reference or correction values for any set of item compare elements 1741 which are part of an individual comparison and are in a state which indicates a data break.

At page 112, line 13– page 113, line 2, please substitute the following paragraph:

Figure 51 describes a range of features which support the modification of client related details through the application's user interface. These features are accessible to users of the application via the file and client details menu options in 5101. On selecting these options, the system utilizes the interface programs 1105 – 1111 to retrieve the client object 1211 for the user and display the current primary and secondary information for the related client in 5102. The user is then provided with a plurality of options: (a) the users may modify the name of the client in 5103; (b) the user may change the client header detail string which is used at the beginning of each data input template record to identify the client, specify the user identifier used for data

feeds, and provide formatting of other required system information such as business date in 5104; (c) the use may modify the client side directories which are used to determine where on a client's PC data is sent to or retrieved from when interacting with the application server in 5105; and (d) the user may modify the server side directories which are used to determine where on server data is sent to or retrieved from for the client in 5106.

At page 114, line 19 – page 115, line 10, please substitute the following paragraph:

After instantiation, a record is created for the object in table rharchdata 2204 in 5302. Once complete, the process retrieves the set of group match queue objects 1641 for the data group 1661 and for each of these group match queue objects, puts XML based headers around the objects data which is retrieved and represent as a XML based string containing the field identifiers and field values for each required field of the object. This computed string is then appended to a string variable that will be saved after all group match queues are processed. Once this iterative process is complete the archive data object GrpMchQueData 1815 is set to the computed string comprising this entire set of group match queue object data in 5303. This extraction and compression process is then performed on the data group's set of data group compare objects 1691. And, once this iterative process is complete the archive data object DatGrpCmpData 1816 is set to the computed string comprising this entire set of data group compare objects object data in 5304. The process then initializes the minimum and maximum business dates and system dates for the archive data object 1801, recitemminbusdate 1817, recitemmaxbusdate 1818, recitemminsysdate 1819, recitemmaxsysdate 1820 using the data group object's 1661 lstbusdatupd 1666 and lstsysdateupd 1667, in 5305.

At page 115, line 11 – page 116, line 12, please substitute the following paragraph:

Next, a set of temporary string variables is created and each initialized to the empty string. The set includes variables for computing archive strings for the set of, original data record strings, reconciliation item archive strings, item information strings, and item compare element strings in 5306. The data group's reconciliation items 1701 are then retrieved and for each the following processes are completed. First, the archive groups minimum and maximum business dates are adjusted based on the items busdate 1704 and sysdate 1705, in 5309. Second, the item original text string, item 1703 is enclosed in XML headers and appended to the variable for this data in 5310. Third, the reconciliation items archive string is retrieved, using the item's toarchstring method, enclosed in XML headers and appended to the related variable, in 5311. Fourth, the set of related item information element's 1731 is retrieved and each of the objects is converted to an archive string, enclosed in XML headers. When the entire set has been processed the resulting string is enclosed in a further set of XML headers indicating which reconciliation item the data belongs to. Then this entire string is appended to the related variable in 5312. Fifth, the set of related item comparison element's 1741 is retrieved and each of the objects is converted to an archive string, enclosed in XML headers. When the entire set has been processed, the resulting string is enclosed in a further set of XML headers indicating which reconciliation item the data belongs to. Then this entire string is appended to the related variable in 5313. Once all reconciliation items 1701 have been processed the recitemorigtext 1821 is set using the related computed string value in 5314, the recitemdata 1822 is set using the related computed string in 5315, the reciteminflmntdata 1823 is set using the related computed string in 5316, and the recitemcmplmntdata 1824 is set using its related computed string in 5317. On completing this process the data group object 1661 is deleted from the system which deletes all

related child objects as indicated in figure 3116 as shown in figure 31, in 5318. At step 5319, if no errors have occurred then the process ends with step 5321; otherwise, an exception is thrown in step 5320 and the effects of the entire process are reversed.

At page 116, line 13 – page 117, line 2, please substitute the following paragraph:

Figure 54 describes the process for restoring an archive data group 1801 from the application's online archive back to the production environment. This process begins with the receipt of the related call containing the identifier of the archive data group object 1801, in 5401. The process then retrieves the archive data object 1801 for the identifier, the base object 1201 for the application, the client object 1211 for the archive data object's client identifier 1803, the reconciliation object 1341 for reconciliation identifier 1804, in 5402. The reconciliation data object 1601 is retrieved or created using the key identifier 1805 and other related information from the archive data object 1801, in 5403. A new data group object 1661 is added to the reconciliation data object 1601 and in a series of calls the new data group's notes field 1671, has error field 1672, error message field 1673, number of systems unmatched (noossysunmched) 1668, group stat (grpstat) 1669, each have their values set using the related archive data object variables in 5404, 5405.

At page 117, line 3 – page 118, line 4, please substitute the following paragraph:

The restore from archive process increments the correct status counter on the reconciliation data object 1601, one of the fields 1610 – 1616, the selection of which is based on the group status (grpstat) 1669, in 5406. The process retrieves the data group compare data (datgrpcmpdata) 1816 and for each sub record in this string, a data group compare object 1691 is

added to the data group with the appropriate compare identifier 1693 and compare status 1694 in 5407. Then, in a series of calls, the process will retrieve the set of reconciliation item original text stings 1821, reconciliation item data 1822, reconciliation item compare element records 1824, and reconciliation item information records 1823, in 5408. The process then breaks down each of these strings using an internal XML parse routine and for each reconciliation original text string, creates a new reconciliation item 1701 for each string, set the field values of the item using the corresponding reconciliation item data, creates a compare elements 1741 for each of the items corresponding compare data strings, and creates an information element 1731 for each of the items corresponding information data strings 5409-5412. During this process, the group identifiers are updated on each of the reconciliation items 1701 using the new data group's identifier values 1665, 1670 and the system dates for both the reconciliation data object 1601 and the data group object 1661 are updated. After processing all reconciliation item sets, the application restores the set of system match queue objects 1631 and the set of group match queue objects 1641 from the related archive string's grpmchquedata 1815. This is achieved by using the reconciliation data object 1601 to retrieve or create a system match queue 1631 for each of the system identifiers in the string and then for each system match queue using the system match queue 1631 to add the group match queue object 1641 for the new data group object 1661, in 5413. After successfully completing these steps, the archive data object 1801 is deleted form the application in 5414 and if no errors have occurred in step 5415, the process ends with step 5417. If any errors do occur during the process, a restore from archive exception is thrown back to the caller in 5416, the effects of the process are reversed, and the process terminates with step 5417.

At page 118, lines 5 - 19, please substitute the following paragraph:

Figure 55a-b describes range of features available through the applications user interface for managing both archive processing and archive data. This figure is divided into two sections, each of which is accessible through related screens in the application's user interface. In figure 55a, in step 5501, the process of utilizing functionality for running the archive process for selected reconciliation begins. This facility is accessible through the utilities menu and the archive options menu in 5501. The application utilizes the adapter program 1162 – 1164 to retrieve and display the set of archive move control objects 1521 for the related client object 1211, in 5502. On completing the display process, the user may select a given control and initiate its archive data process. The archive data process retrieves the control objects archive move status objects 1541 and uses these in conjunction with the derived business or system cutoff date to retrieve a set of data group objects 1661, which it will attempt to archive. For each object in the set, the process calls the move data group to archive process of figure 53. Once all data groups in the set are removed, the process retrieves the set of reconciliation data objects 1601 for the reconciliation and deletes any of these objects having no remaining child data group objects, in 5503.

At page 118, line 20 – page 119, line 13, please substitute the following paragraph:

In Figure 55b, step 5504 begins the process of utilizing the applications features for reviewing and restoring the systems archived data groups. These features are accessible to client via the processed data and data archive menu options. Selecting these options will utilize the interface adapter programs 1151 – 1155 to provide the following set of options. On initial selection, the user interface is displayed for the first reconciliation in the list of user reconciliation and the default selection criteria of the related adapter program, in 5505. At any